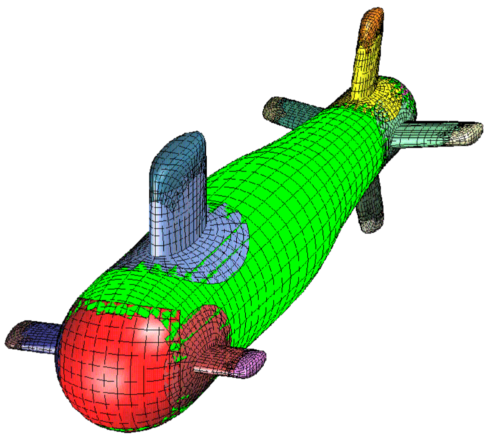
Component Based Hybrid Mesh Generation

Kyle Chand
Centre for Applied Scientific Computing
Lawrence Livermore National Laboratory
Livermore, California
www.llnl.gov/CASC/Overture

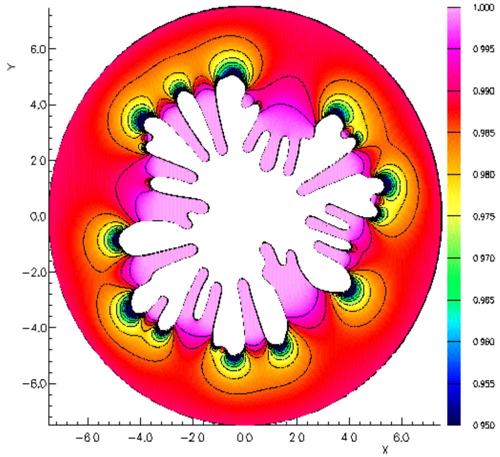
Overture team: David Brown, Kyle Chand, Petri Fast,
Bill Henshaw, Brian Miller, Anders Petersson,
Bobby Phillip, Dan Quinlan

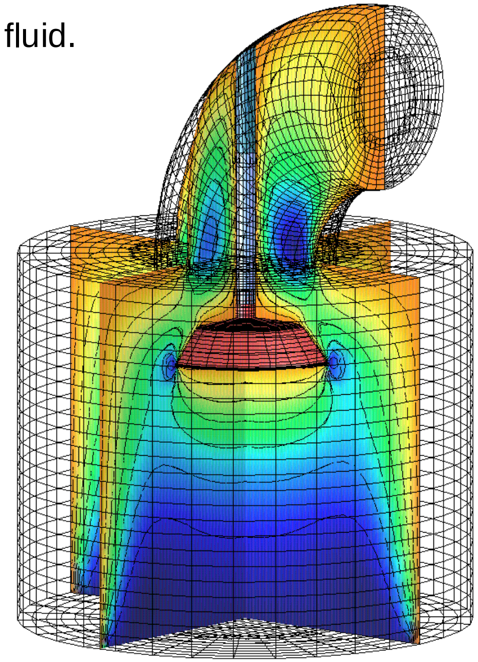
Overture: A Toolkit for Solving PDEs



Overlapping Grids
Fig: Bill Henshaw

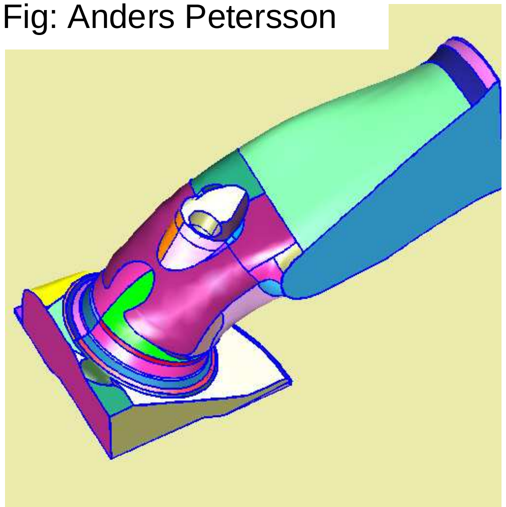
Hele Shaw flow of a non-Newtonian fluid.
Fig: Petri Fast.

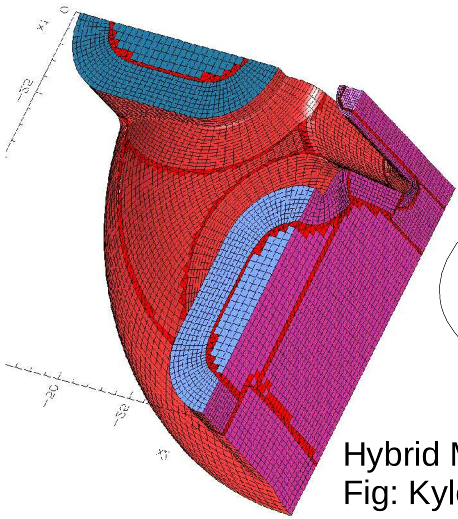




Moving Piston, Incompressible Navier-Stokes
Fig: Bill Henshaw.

CAD Geometry
Fig: Anders Petersson





Hybrid Meshes
Fig: Kyle Chand

PDE Solver Development

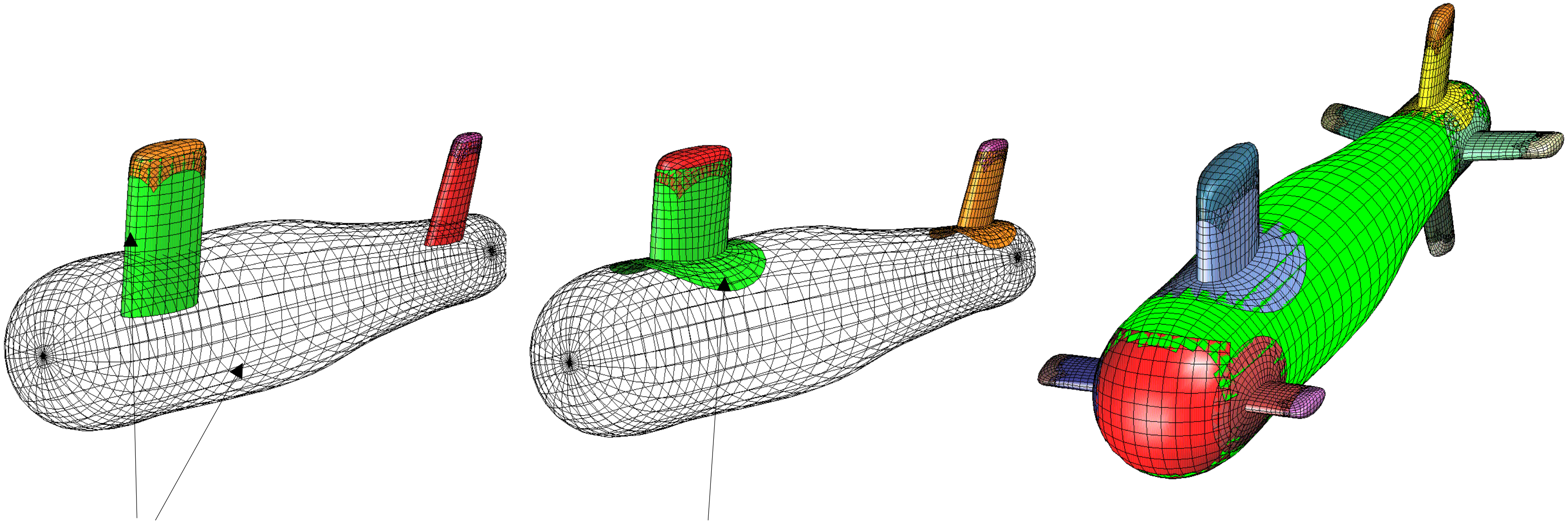
Grid Generation

Geometry

Component Based Grid Generation

Component based grid generation: build structured overlapping component grids and connect as overlapping or hybrid grids.

Ogen : automatic generation of an overlapping grid, given overlapping component grids.

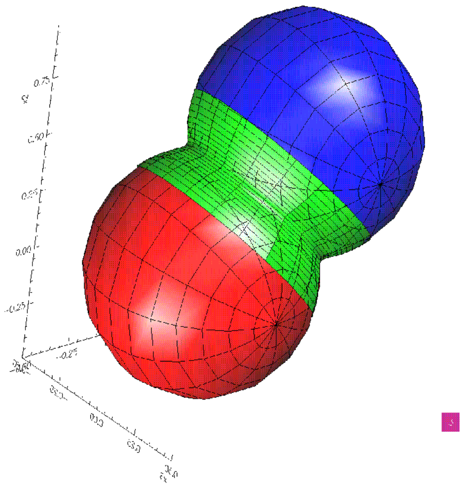


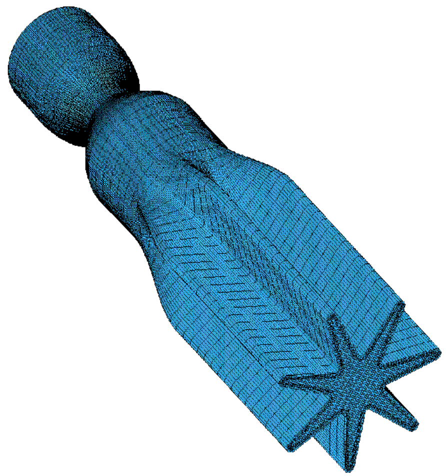
Build overlapping components

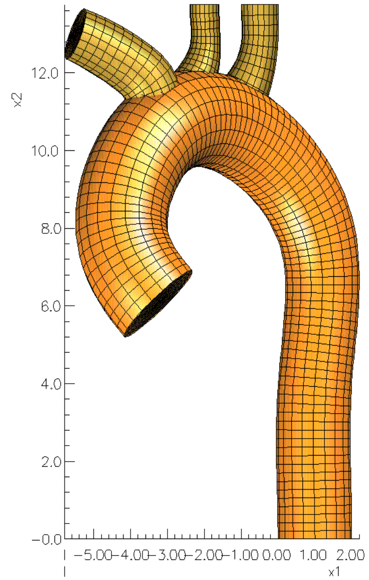
Join components at boundaries.

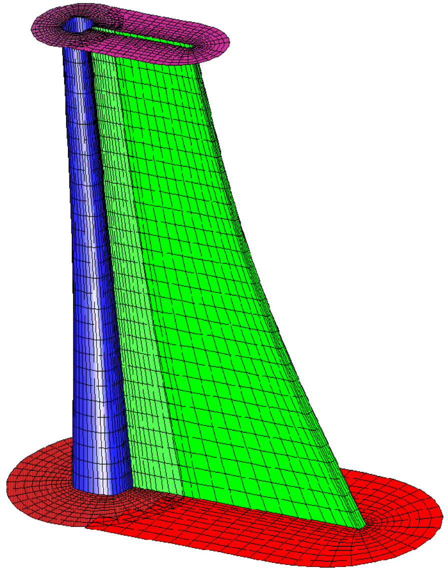
Compute the overlapping or hybrid grid

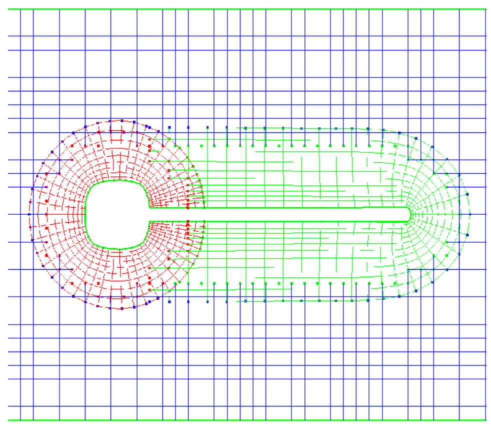
Component Mesh Generation in Overture





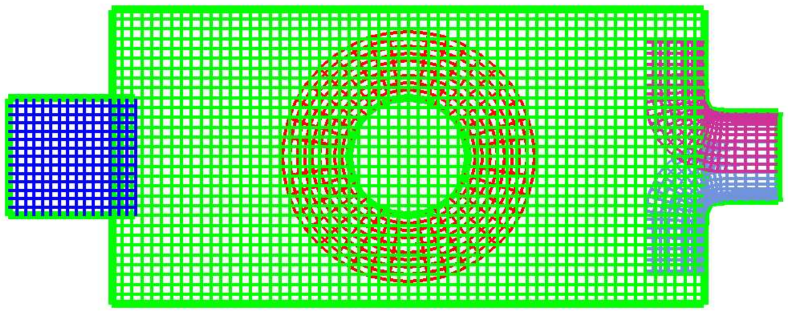




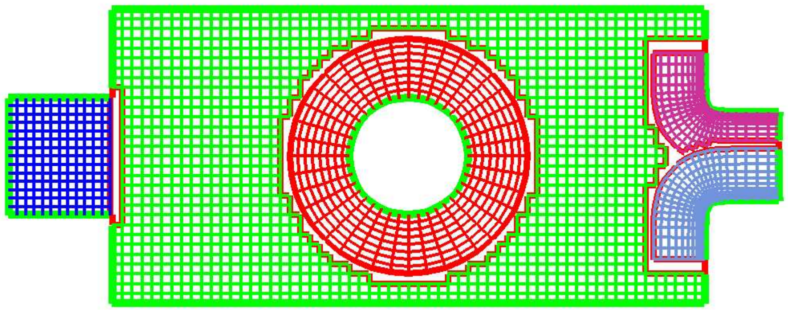


Hybrid Mesh Generation

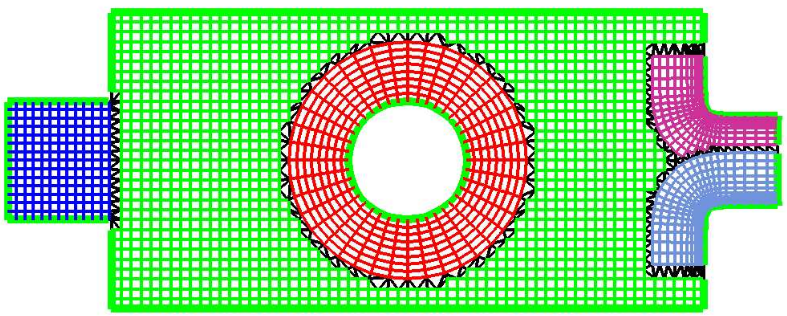
Hybrid meshes connect structured grids
with unstructured mesh.



Component grids



Overlap removed



Hybrid mesh

Hybrid Mesh Algorithms and Software

- Overture Mapping classes --> component grids
- Overture Overlapping grid generator --> automatic hole cutting
- 2/3D Advancing front unstructured mesh generator
- UnstructuredMapping container class for the mesh
- Mesh optimization algorithms

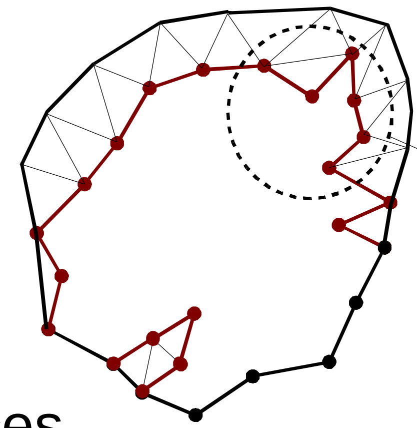
Similar work :

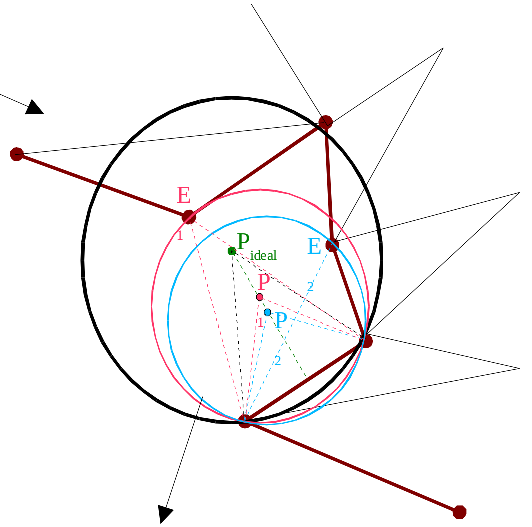
- Liou, Zheng and Civinskas --> DRAGON grids (1994)
- Shaw, Peace, Weatherhill (1994)
- Weatherill gives a general discussion in *Numerical Grid Generation in CFD '88*

Advancing front sources :

- Lo (1985,1991)
- Peiró, Peraire, et al. (1987, 1992, ...)
- Löhner (1988, 1996)
- George, Seveno (1994)
- Jin, Tanner (1993)

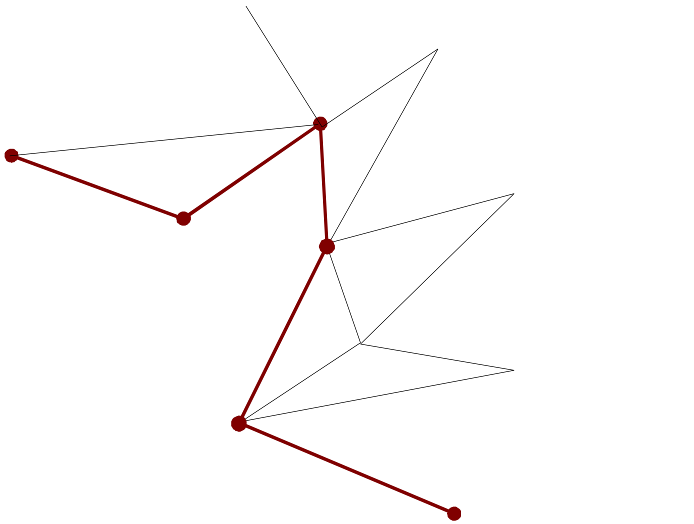
Advancing Front Algorithm





Begin with an initial front
- line segments
- triangles/quads

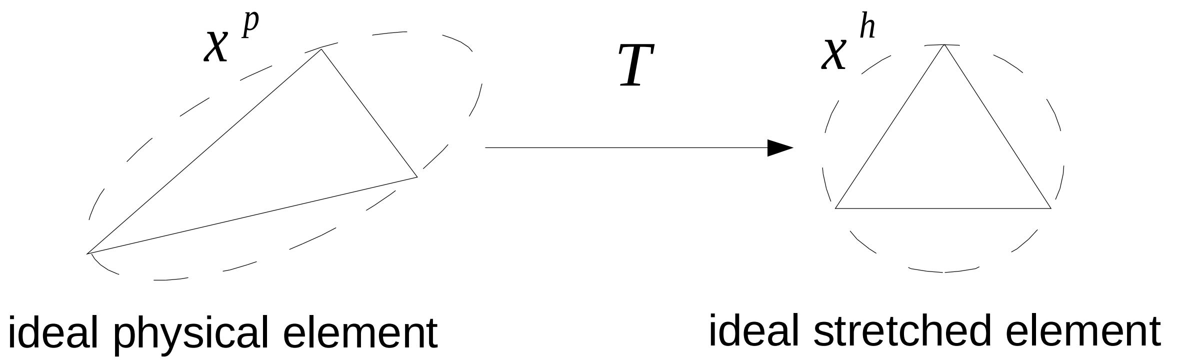
Select a face to advance
- search for existing vertices
- create candidate new vertices
- prioritize candidate elements
- select the first "consistent" element
  - no intersections
  - no enclosed vertices



Delete old face(s) from the front
Add any new faces
Repeat until the front is empty

Mesh Spacing Control

$$x^h = T\,x^p$$

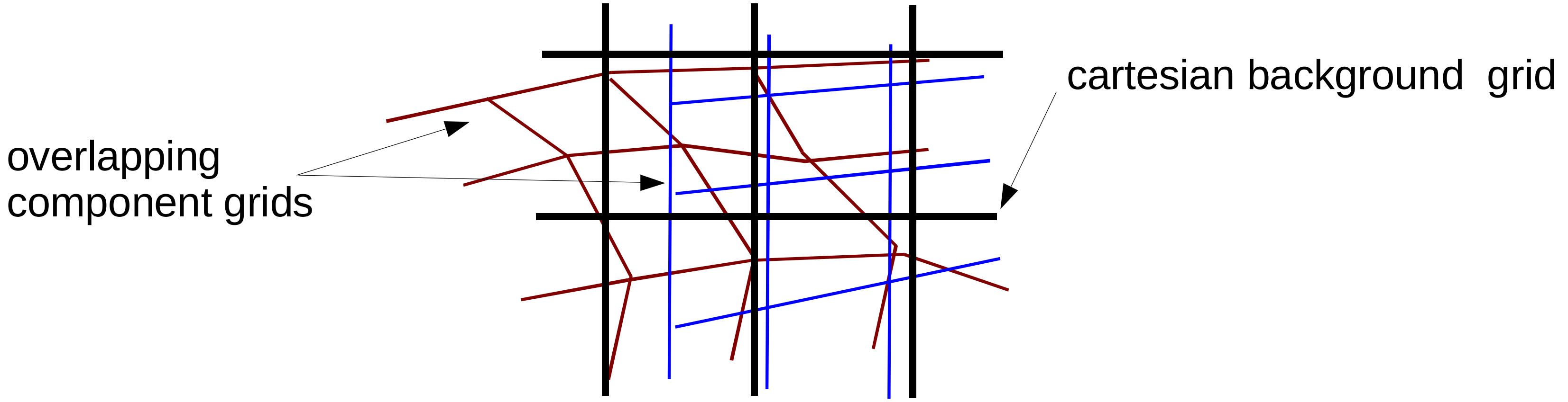


Points in the viscinity of the advancing face are mapped using T

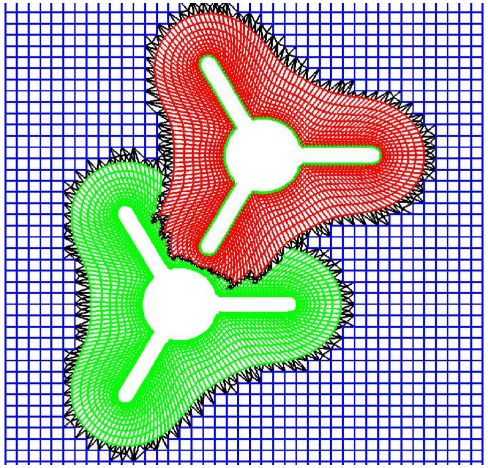
The algorithm attempts to make a new element as equilateral as possible

T is computed by averaging the grid Jacobians from the overlapping grids

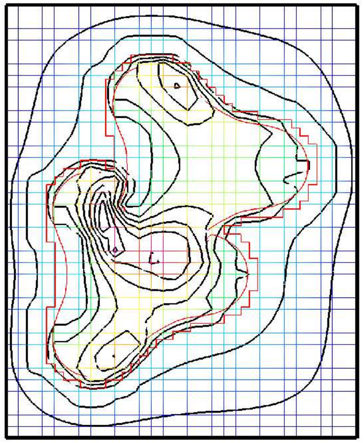
(A Jacobi iteration of the elements of T smooths the stretching function)



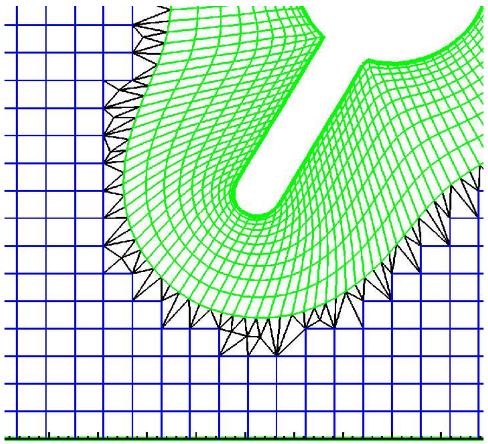
Mesh Spacing Control

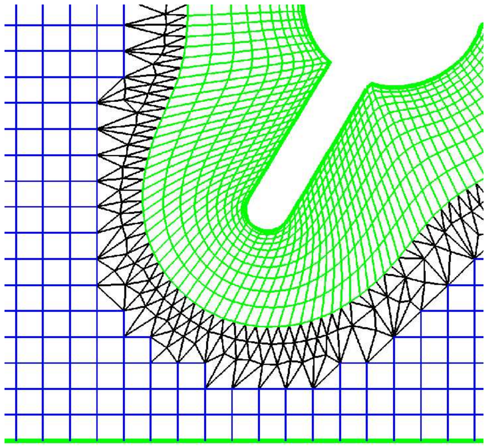


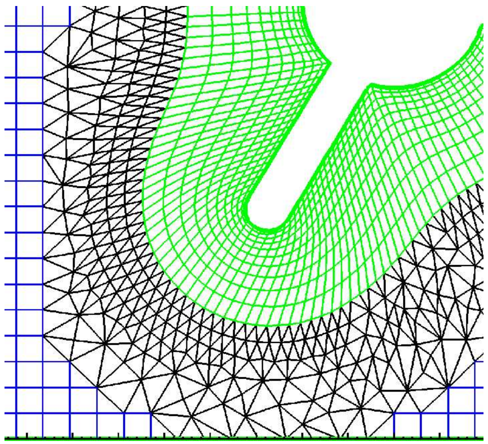
Unstructured mesh blends the spacings of the component grids



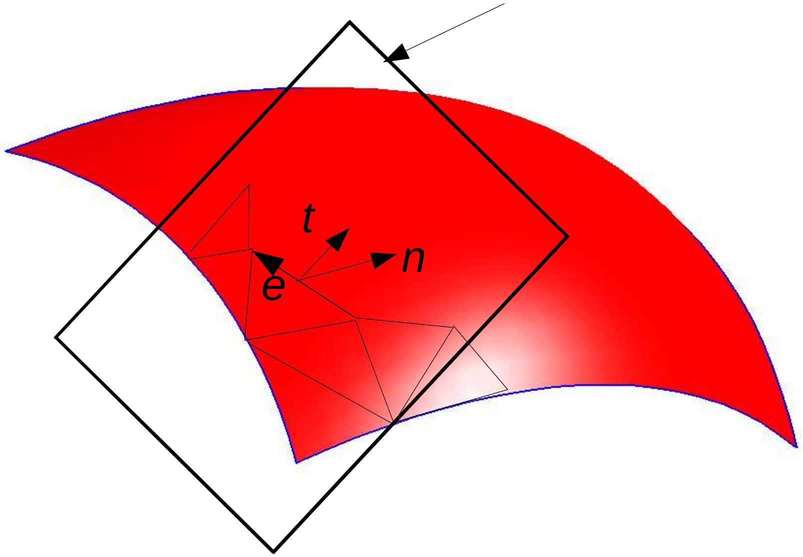
A background cartesian gr
stores stretching informati
from the original compone...
grids







Surface mesh generation



e = edge vector pointing along the front
n = surface normal at midpoint of edge
t = advancement direction

$$t = e \times n$$

$$P^h_{ideal} = P^h_{midpoint} + d\,Tt$$

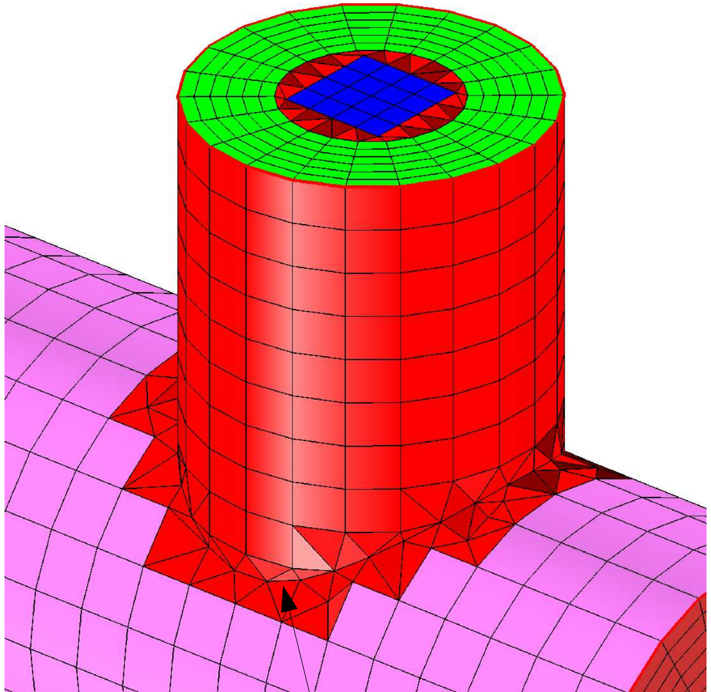
Surface normal n is computed from the geometry at the midpoint of the advancing face.

Points in the neighborhood of the advancing face are transformed by *T* and projected onto the plane defined by e^h and P^h_{ideal}.

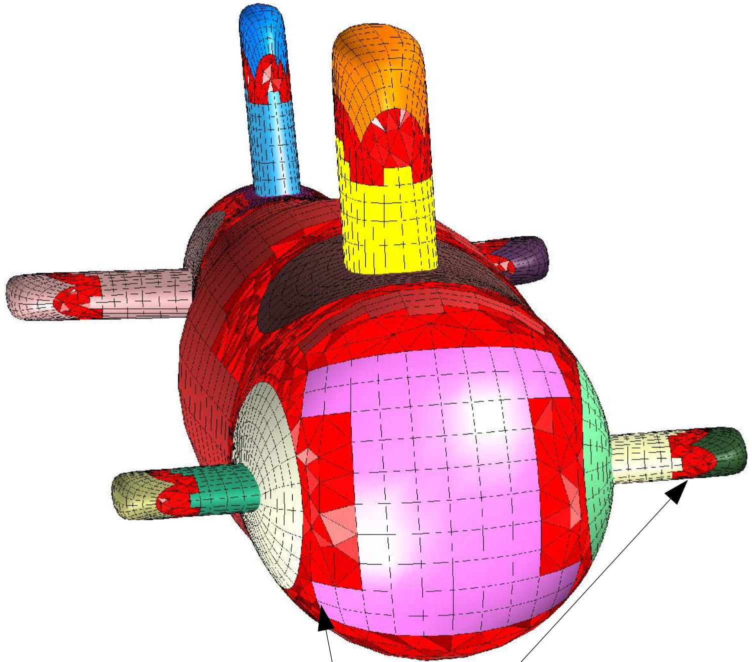
Validity tests are performed in the plane, essentially a 2D advancement.

High curvature surfaces are tricky:
during intersection checks, ignore faces that have surface normals differing by more than (say) 60 degrees from the normal at the current face midpoint.

Surface mesh generation

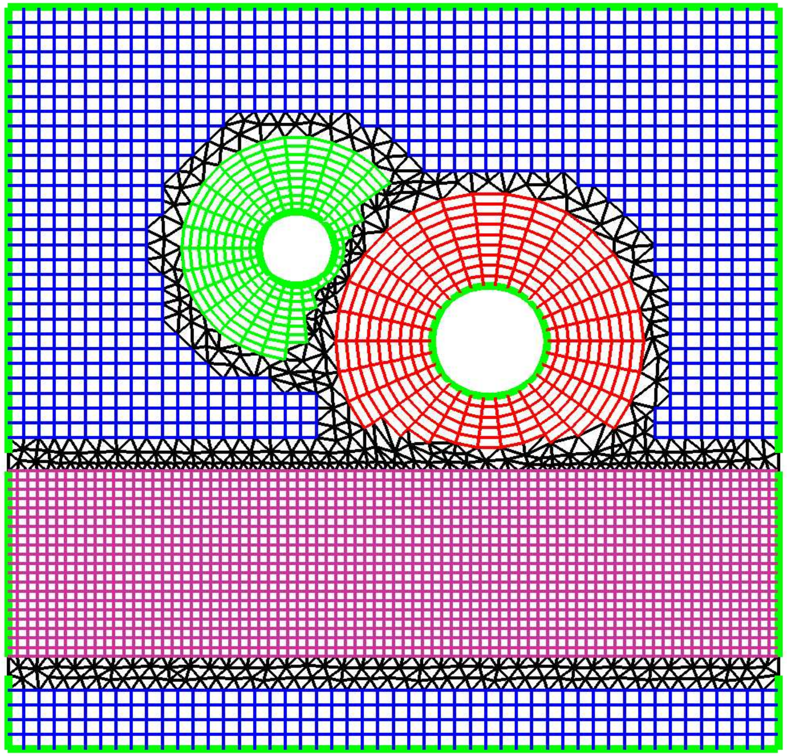


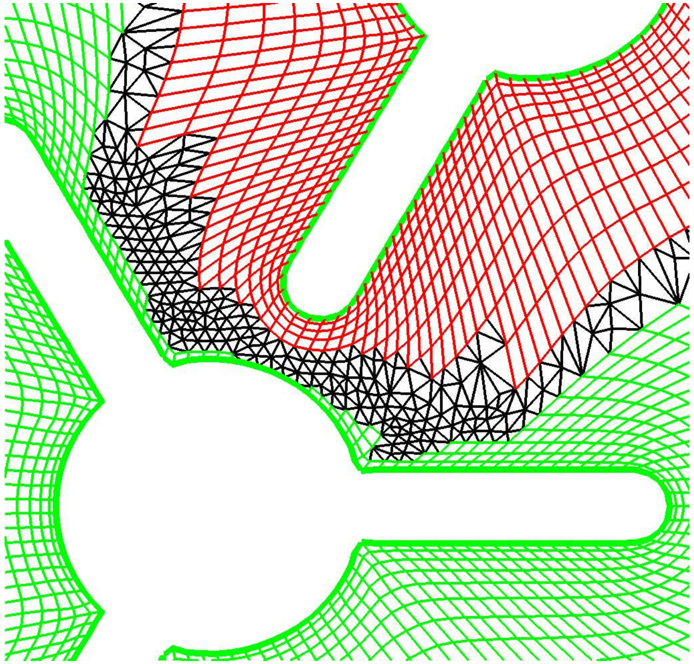
Intersecting surfaces



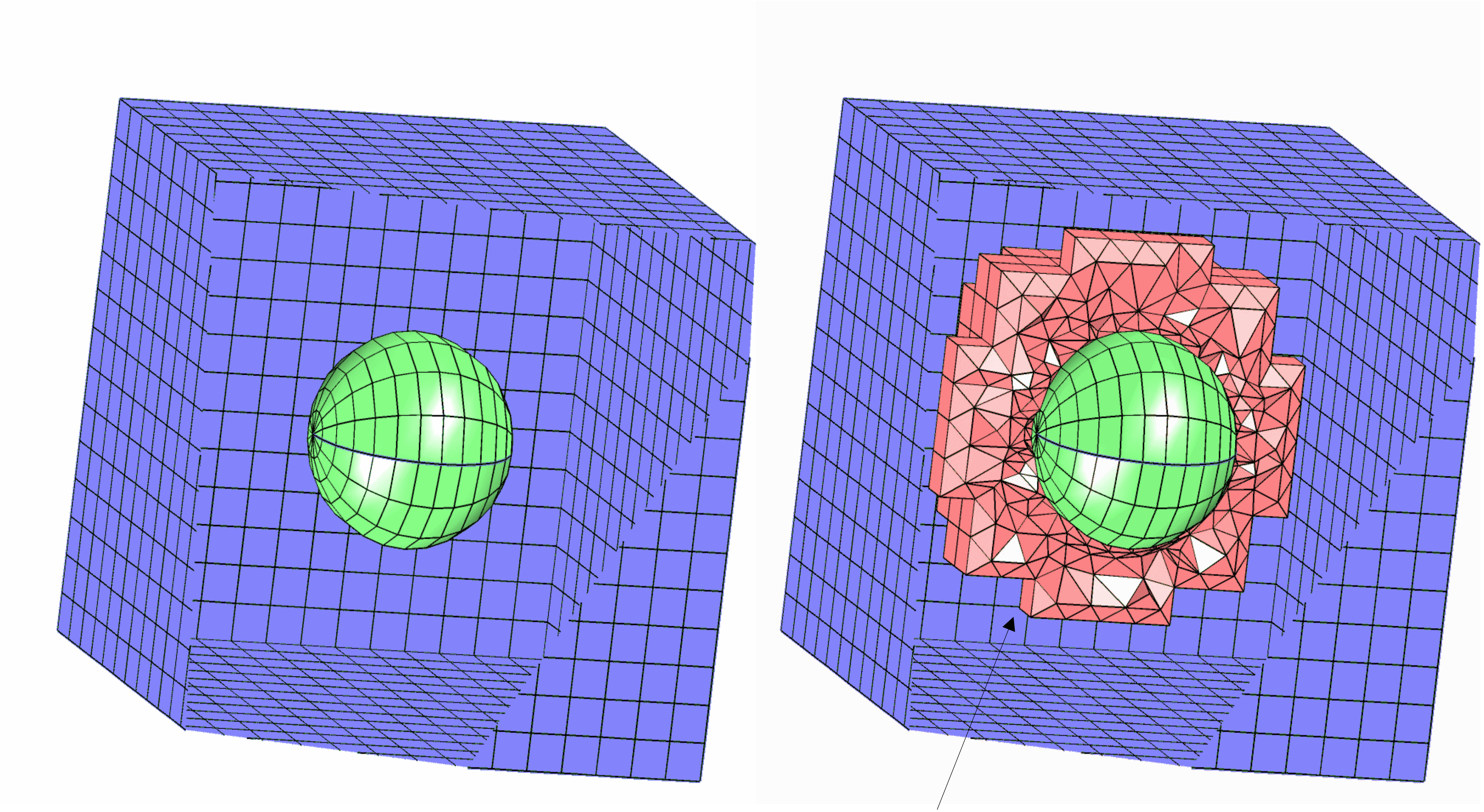
Overlapping surface grids

2D Examples



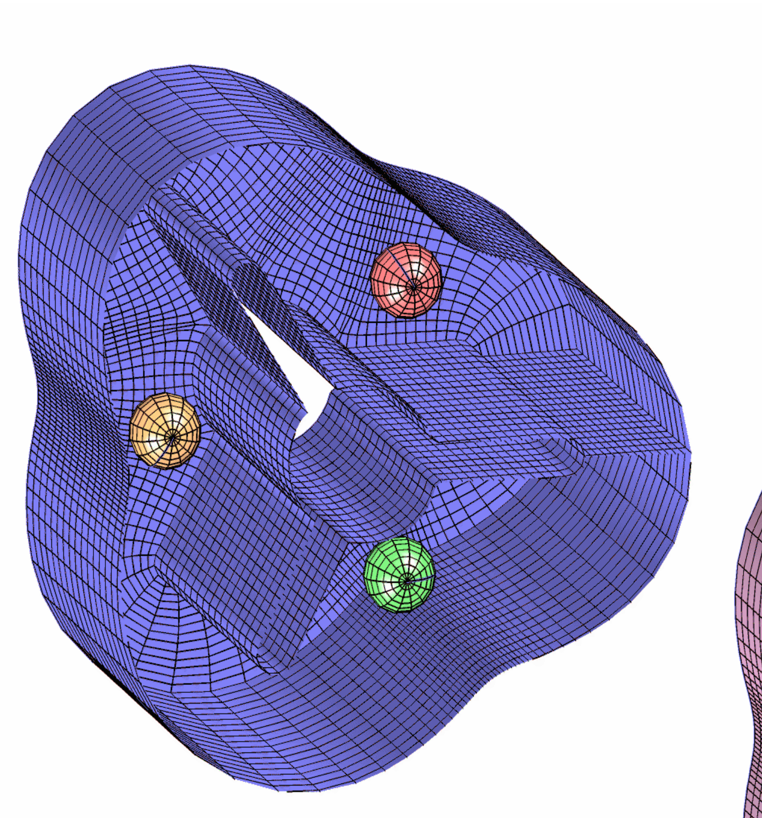


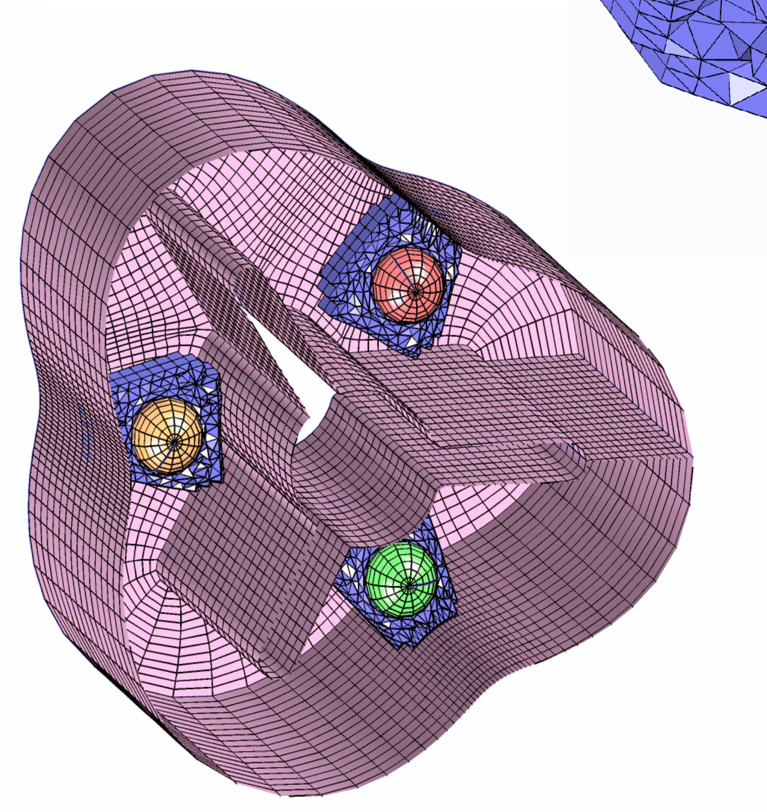
3D Demonstrations

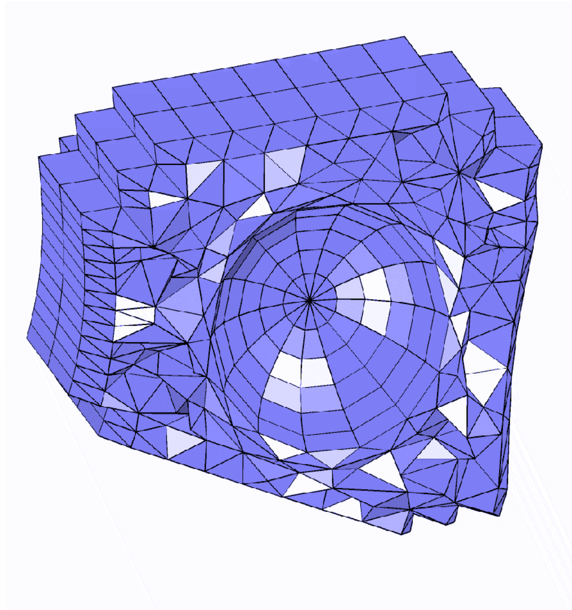


Pyramids interface structured hexes with unstructured tets

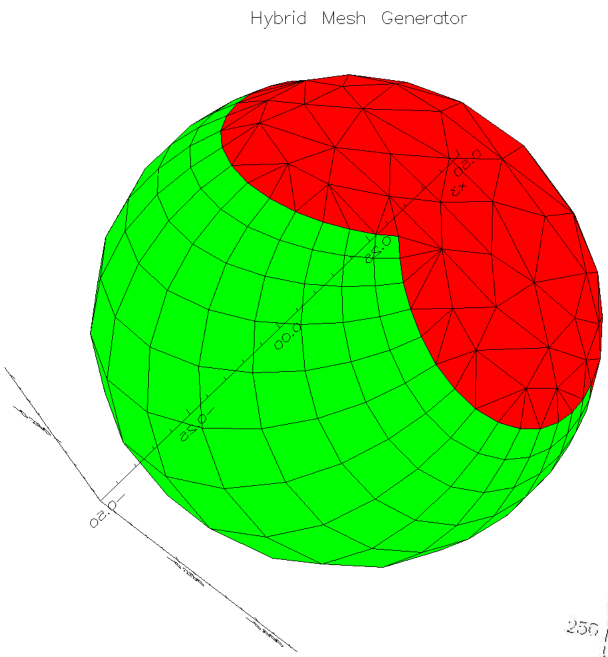
3D Demonstrations

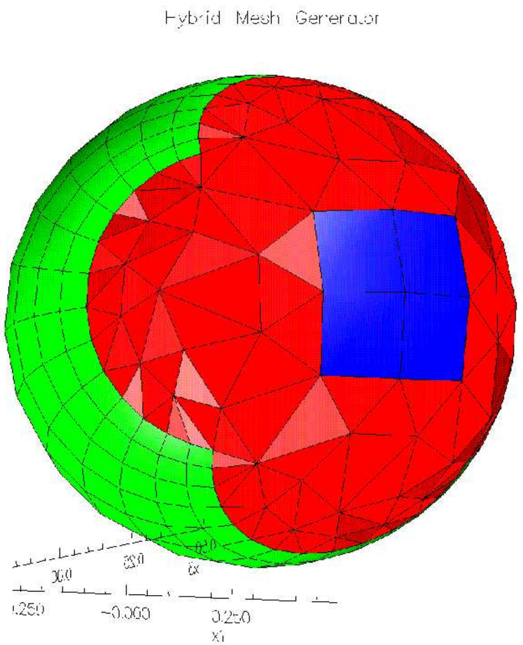


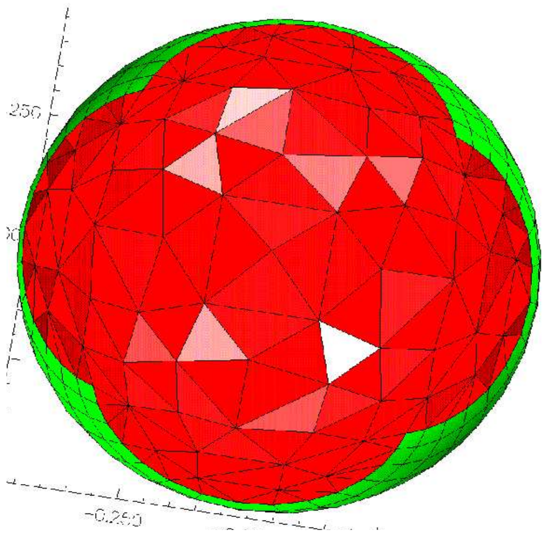




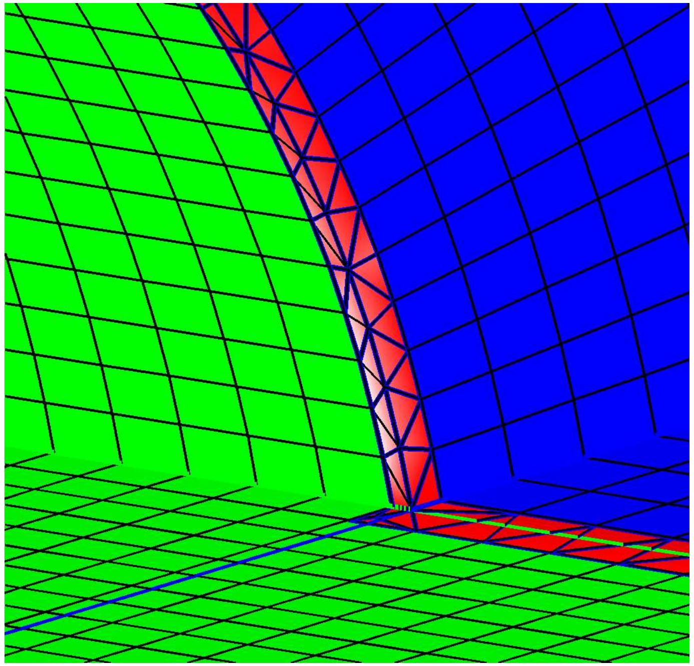
3D Demonstrations

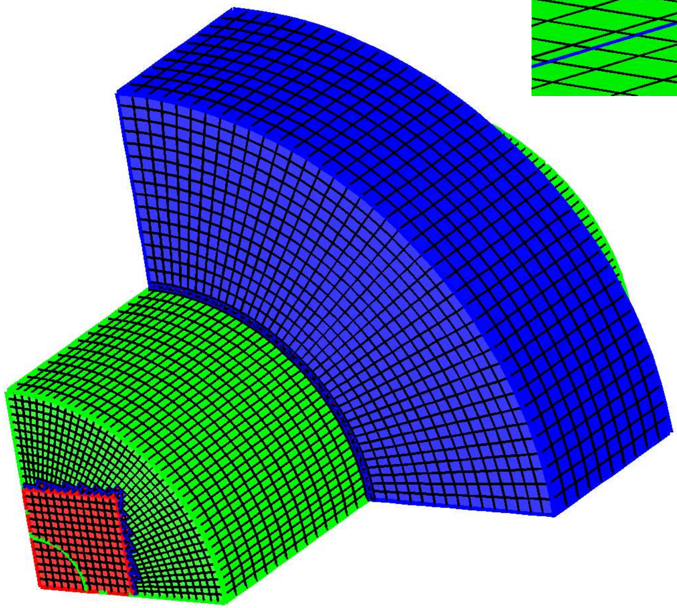


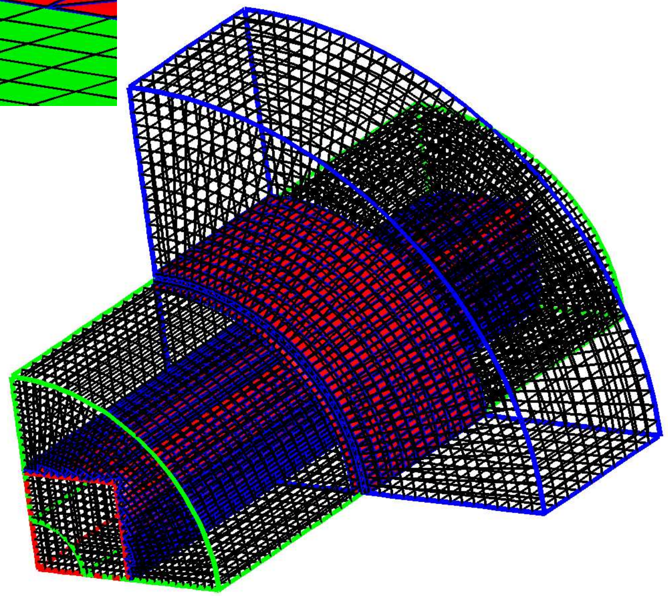




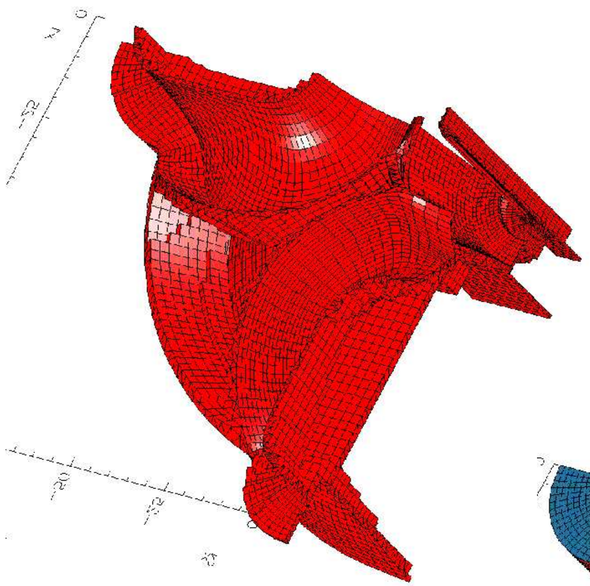
3D Demonstrations

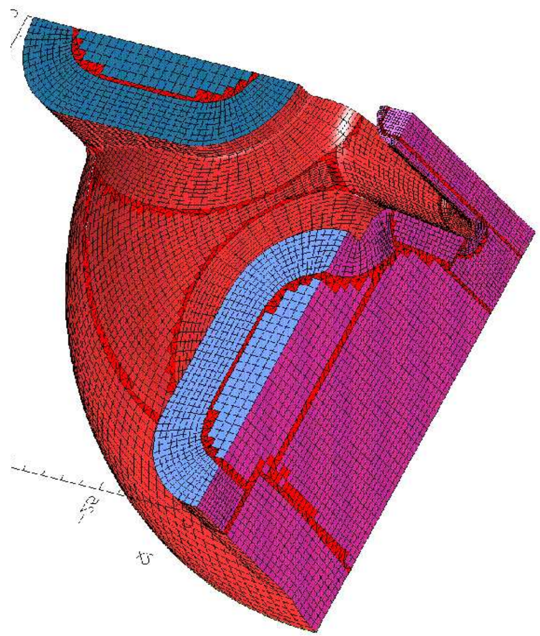


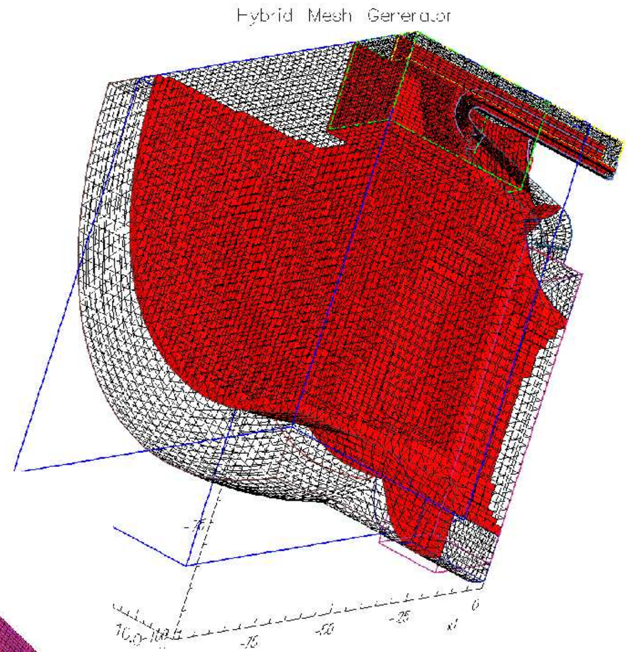




3D Demonstrations



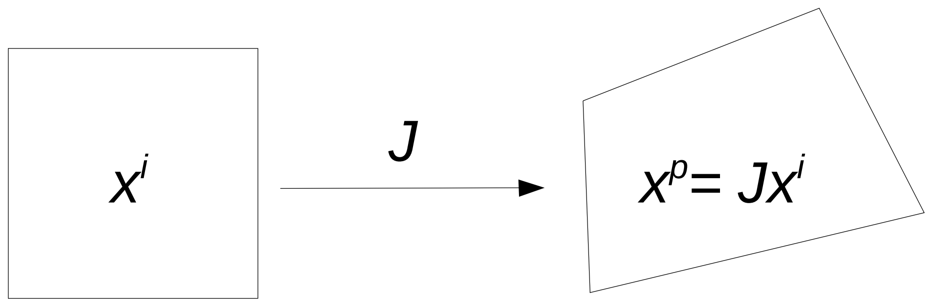




Mesh quality

Mesh quality assessement based on Pat Knupp's Algebraic Mesh Quality metrics (Knupp '99).

Metrics use properties of the Jacobian of the (linear) mapping between the actual and the "ideal" element:

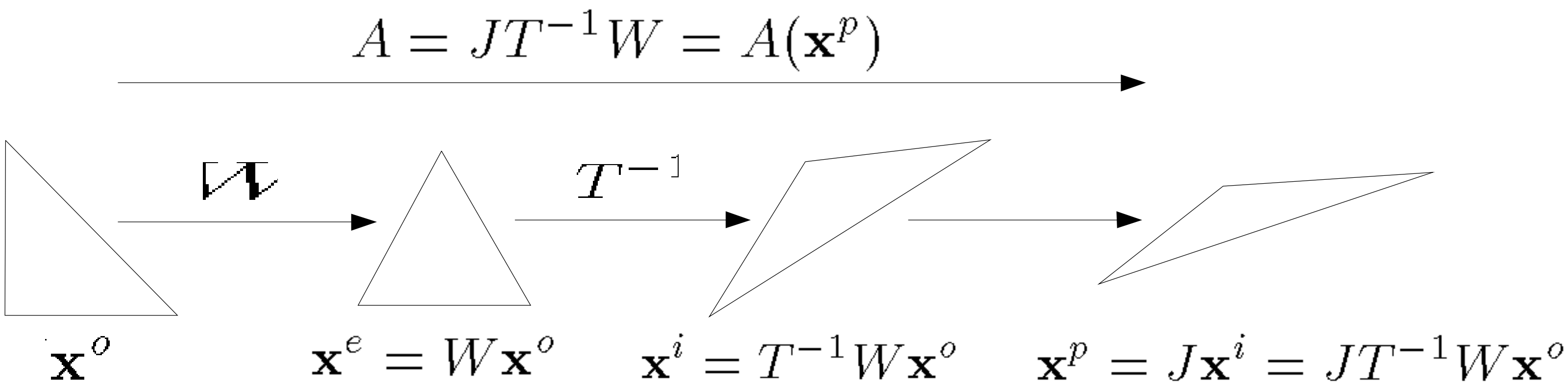


Useful metrics include :

- det(J) – scaled size
- K(J) - Condition number or C/K(J) - "shape" metric
- min(det(J), 1/det(J)) C / K(J) – combined shape and size metric

Mesh quality

Computing the Jacobian between the "ideal" element and the actual element (Pat Knupp) :



$$J = AW^{-1}T = AM$$

W is determined by the shape of the element, *T* by interpolation from the spacing control grid and *A* from the actual element vertices

Element Jacobian calculation

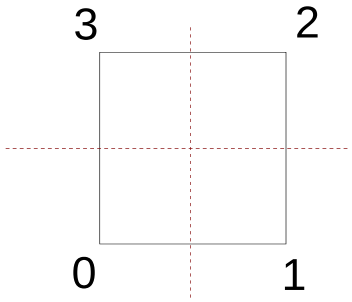
$$A^{tri} = \begin{bmatrix} x_1 - x_0 & x_2 - x_0 \\ y_1 - y_0 & y_2 - y_0 \end{bmatrix}$$

Same as Pat for triangles and test

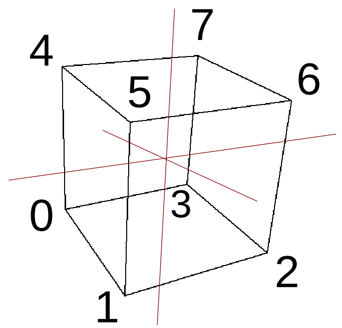
$$A^{tet} = \begin{bmatrix} x_1 - x_0 & x_2 - x_0 & x_3 - x_0 \\ y_1 - y_0 & y_2 - y_0 & y_3 - y_0 \\ z_1 - z_0 & z_2 - z_0 & z_3 - z_0 \end{bmatrix}$$

Centered finite difference to compute the derivatives for quads and hexes:
(note that det(J)>0 for "slightly" tangled quads and hexes...)

$$A^{quad} = \frac{1}{2} \begin{bmatrix} x_1 + x_2 - x_0 - x_3 & x_2 + x_3 - x_0 - x_1 \\ y_1 + y_2 - y_0 - y_3 & y_2 + y_3 - y_0 - y_1 \end{bmatrix}$$



$$A^{hex} = \frac{1}{4} \begin{bmatrix} x_{2367} - x_{0154} & x_{0374} - x_{1265} & x_{4567} - x_{0123} \\ y_{2367} - y_{0154} & y_{0374} - y_{1265} & y_{4567} - y_{0123} \\ z_{2367} - z_{0154} & z_{0374} - z_{1265} & z_{4567} - z_{0123} \end{bmatrix}$$



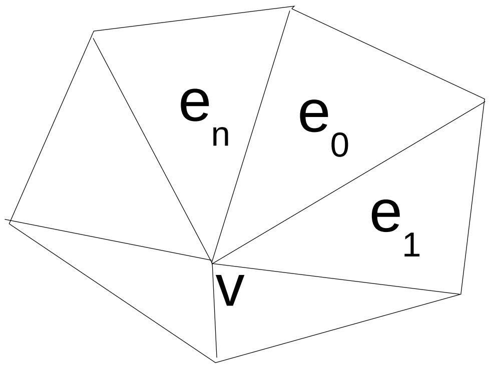
Mesh optimization

Local mesh improvement based on nonlinear optimization of vertex locations (Lori Frietag, Pat Knupp '99, '00, ...)

Define : $f_v = f(x_v) = f(J_0(x_v), J_1(x_v) , ..., J_n(x_v))$
= the objective function at vertex v ($J_e = A_e M_e$)

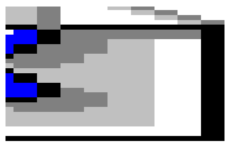
Given a search direction d, iteratively search for an optimal step size using a quadratic line search

$$\mathbf{x}_v^{n+1} = \mathbf{x}_v^n + \mathbf{d}$$



Steepest Descent :

$$f_v(\mathbf{x}_v) = \sum_{e=0}^{n} f_e(J_e(\mathbf{x}_v)) = \sum \kappa_e^2$$



$$\frac{\partial f_e}{\partial x_v} = tr\left(\frac{\partial f_e}{\partial A}\frac{\partial A}{\partial x_v}^T\right) \longrightarrow \mathbf{d} = -d\nabla f_v$$

Mesh optimization

Newton (2D only for now):

compute gradient and Hessian using finite volume approximiation around v
Then:

$$\begin{aligned}\mathbf{d}^0 &= -\left(\frac{\partial^2 f_v}{\partial \mathbf{x}_v^2}\right)^{-1} \nabla f_v \\ \hat{\mathbf{d}} &= \frac{\mathbf{d}^0}{|\mathbf{d}^0|} \\ \mathbf{d} &= d\hat{\mathbf{d}}\end{aligned}$$

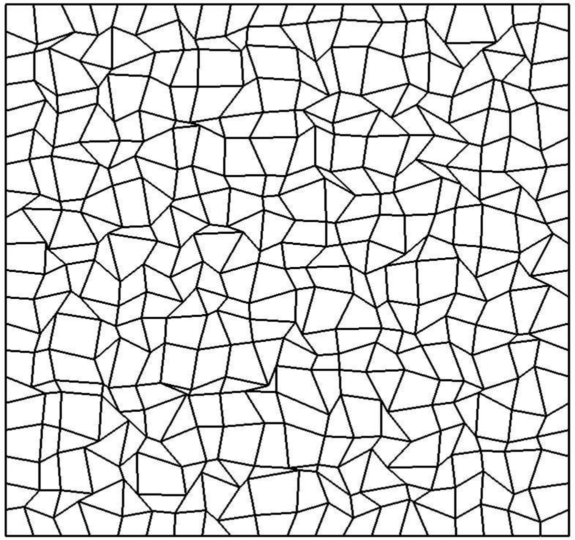
minimize 2 norm of the condition number during the line search:

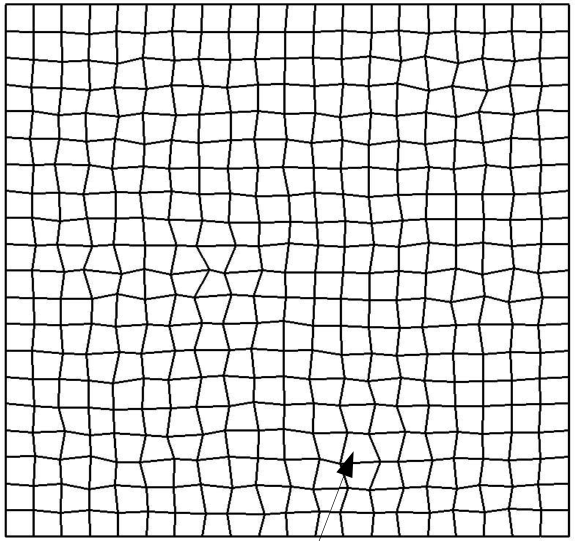
$$f_v = \frac{\sum_{e=0}^{n} \kappa_e^2 det(J_e)}{\sum_{e=0}^{n} det(J_e)}$$

Numerical integration prone to numerical errors when the mesh is very bad (--> nonsymmetric and even negative Hessians!)

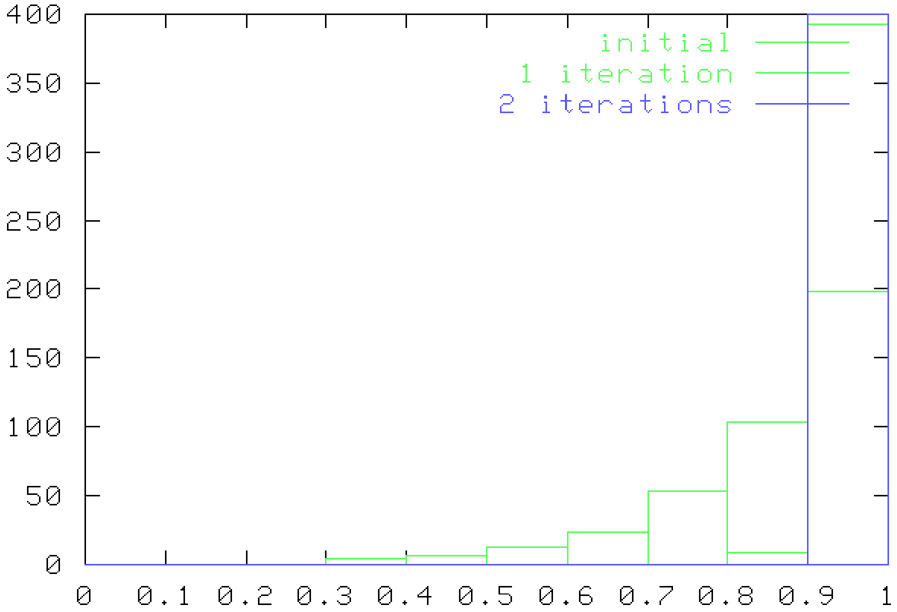
Usefull as second step after a steepest descent step
Still work in progress...

Mesh optimization, preliminary results



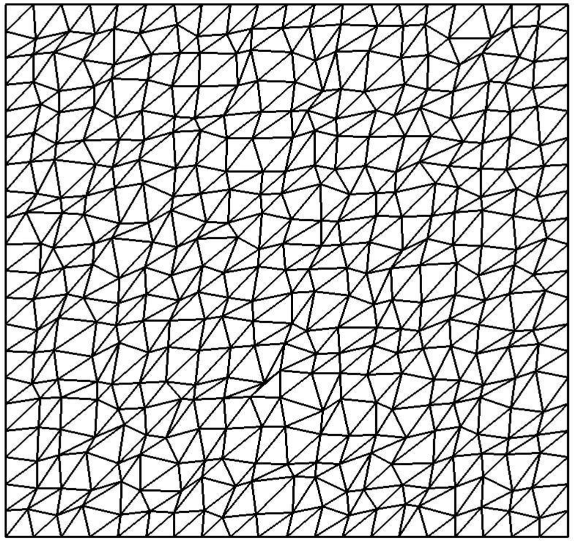


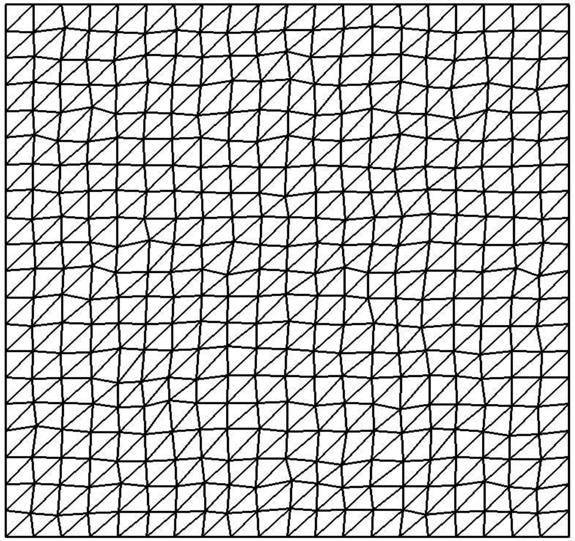
2D quadrilateral optimization test

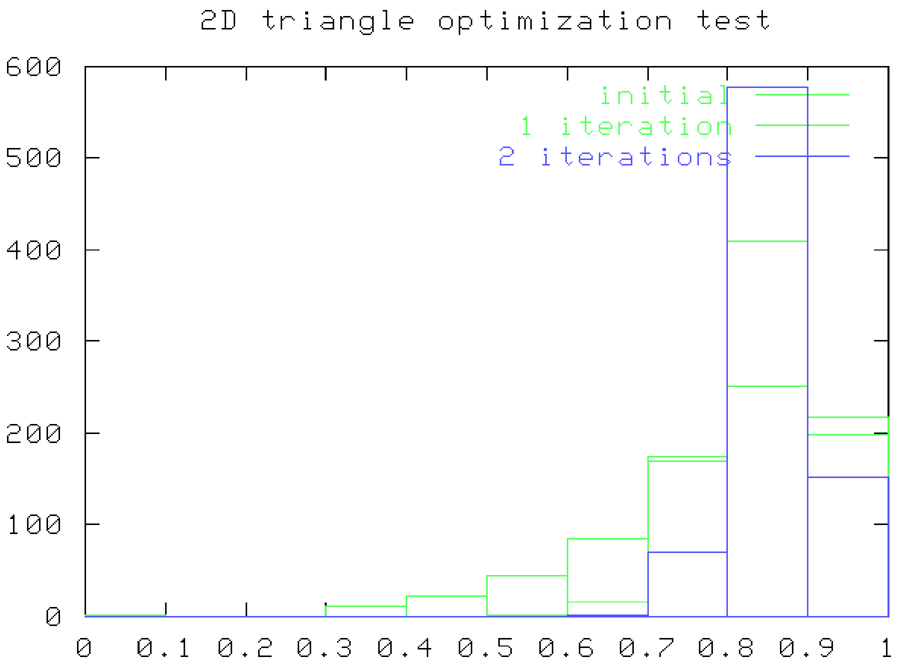


"Hourglass" patterns in the final mesh are due to the cell centered jacobian approximation

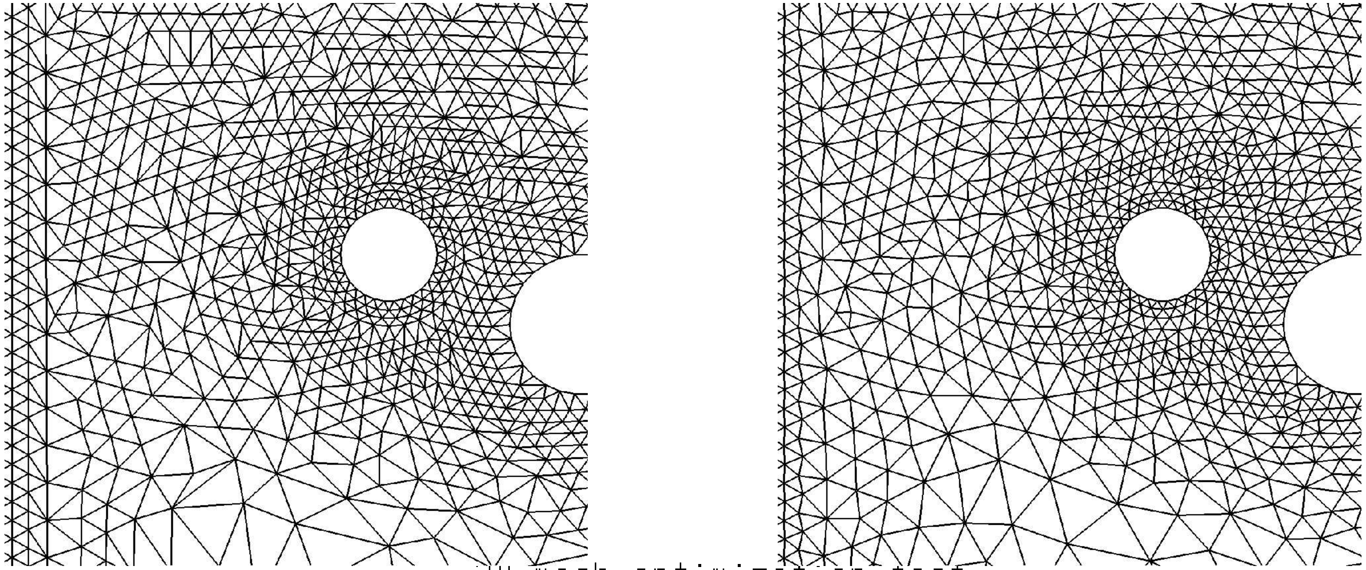
Mesh optimization, preliminary results

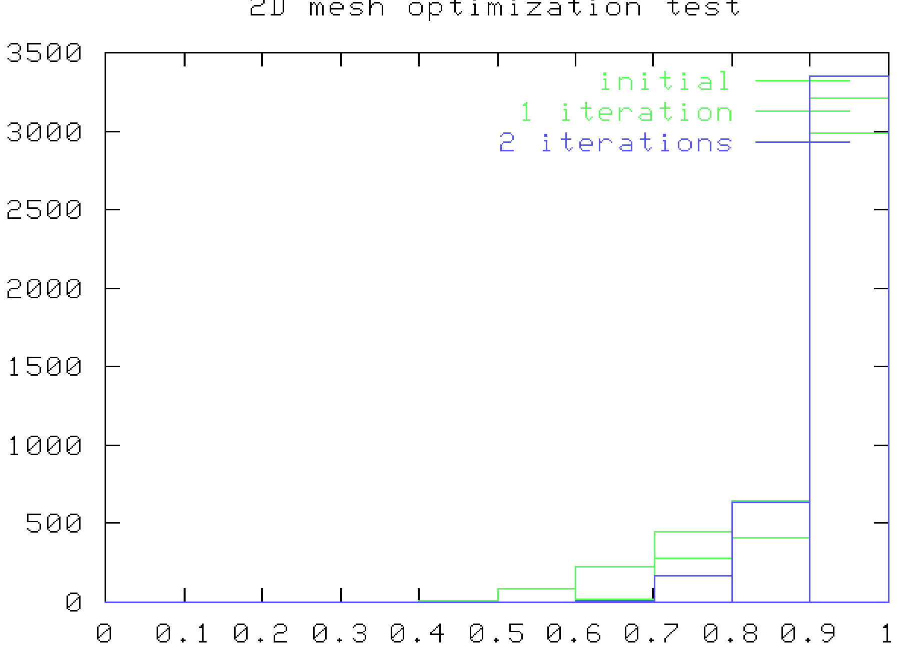




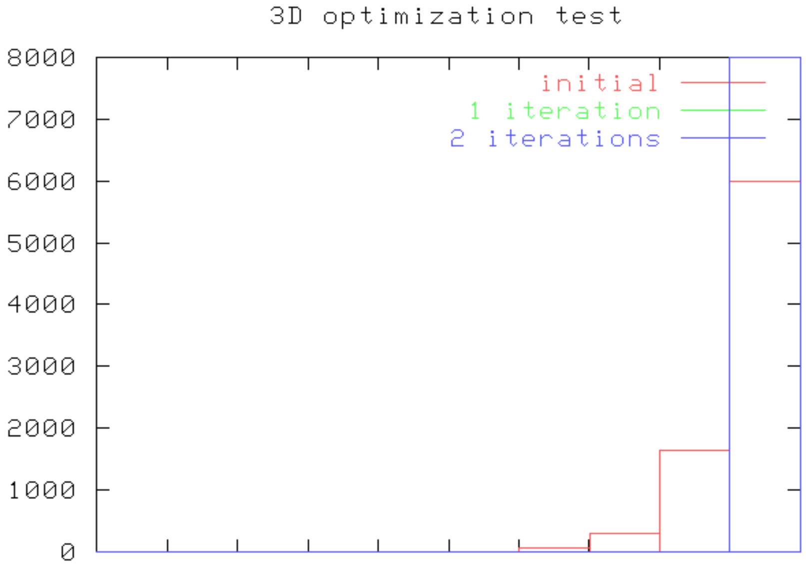


Mesh optimization, preliminary results





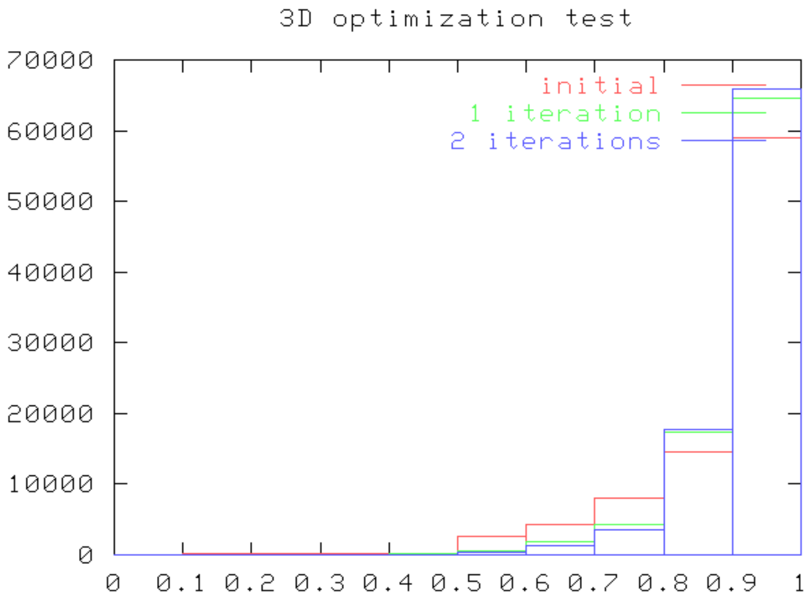
Mesh optimization, preliminary results



Randomized hexahedral mesh

3 iterations

max shape metric: 1.00
min shape metric : 0.97



Pillbox hybrid mesh

3 iterations

max shape metric: 1.00
min shape metric : 0.34

Data Structures and Tools

Geometric search tree (Bonet and Peraire)

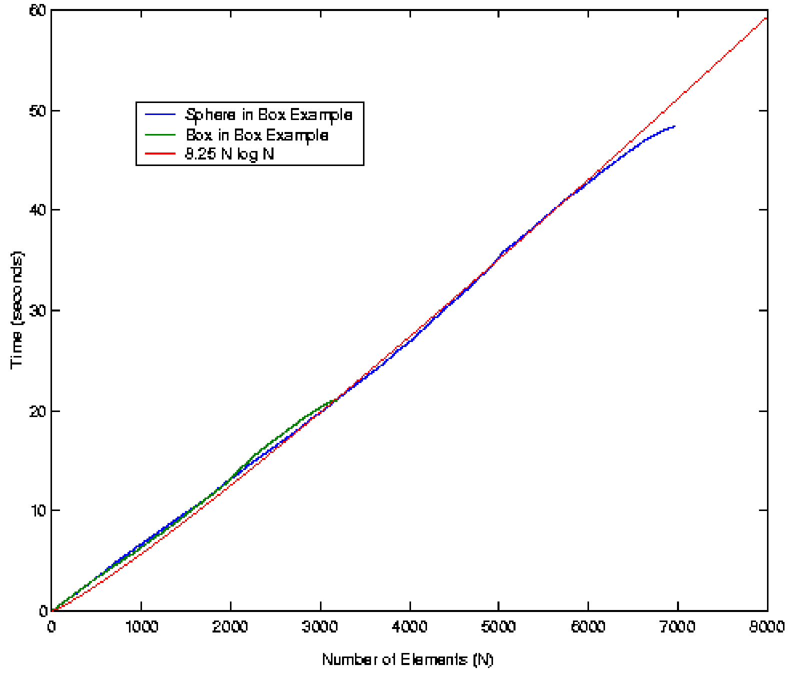
Hash tables and priority queues

Robust geometric predicates

--> Using Jonathan Shewchuk's code

Intersection and orientation tests

O(N logN) scaling where N is the number of elements generated



Current and Future Work

Documentation

Mesh quality is still an issue in 3D :

research and improve mesh optimization tools

- or - use the TSTT interface to Mesquite

automatic hole enlargement prior to mesh generation

Integrate unstructured and hybrid meshes with the rest of the Overture framework (difference operators, solvers, etc)

Obtaining Overture

Overture home page:
www.llnl.gov/CASC/Overture